58

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/928,011	08/10/2001	Holger Sedlak	1999P1177	7290

24131 7590 08/04/2005

LERNER AND GREENBERG, PA
P O BOX 2480
HOLLYWOOD, FL 33022-2480

EXAMINER	
RIZZUTO, KEVIN P	

ART UNIT	PAPER NUMBER
2183	

DATE MAILED: 08/04/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/928,011	SEDLAK ET AL.	
	Examiner	Art Unit	
	Kevin P Rizzuto	2183	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *13 May 2005*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-7* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-7* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. Claims 1-7 have been examined.

2. Acknowledgement of the following papers filed: Amendment as received on 5/13/2005. The papers filed have been placed on record.

Maintained Claim Rejections - 35 USC § 112

3. The following is a quotation of the first paragraph of 35 U.S.C. 112:

> **The specification shall contain a written description of the invention, and of the manner and process of making and using it, in such full, clear, concise, and exact terms as to enable any person skilled in the art to which it pertains, or with which it is most nearly connected, to make and use the same and shall set forth the best mode contemplated by the inventor of carrying out his invention.**

4. Claims 6 and 7 are rejected under 35 U.S.C. 112, first paragraph, as failing to comply with the enablement requirement. The claim(s) contains subject matter which was not described in the specification in such a way as to enable one skilled in the art to which it pertains, or with which it is most nearly connected, to make and/or use the invention.

40. Many factors are to be considered when determining whether there is sufficient evident to support a determination that a disclosure does not satisfy the enablement requirement and whether any necessary experimentation is "undue". [In re Wands, 858 F.2d 731, 737, 8 USPQ2d 1400, 1404 (Fed. Cir. 1988). *See also MPEP § 2164.01(a).]* For example, the factors include but are not limited to:

(A) The breadth of the claims;

(B) The nature of the invention;

(C) The state of the prior art;

(D) The level of one of ordinary skill;

(E) The level of predictability in the art;

(F) The amount of direction provided by the inventor;

(G) The existence of working examples; and

(H) The quantity of experimentation needed to make or use the invention based on the content of the disclosure.

5. The nature of the invention is a computer hardware element for processing instructions from multiple different languages, a complex technology.

6. The state of the prior art shows that multi-language architectures with plural program counters including adding and subtracting units are atypical.

7. The amount of direction provided by the inventor is not sufficient to allow one of ordinary skill in the art to make or use the invention without undue experimentation. Multiple program counters each having the capability to selectively add or subtract an instruction length or to not add or subtract an instruction length to the program counter is not disclosed. Also, multiple program counters being used with the selective adding or subtracting of an instruction length to an offset was not disclosed. It appears Applicants combined two or three different embodiments, one with multiple program counters and one with selectively adding/subtracting an instruction length to a program counter and one with selectively adding/subtracting an instruction length to an offset value. However, this was not disclosed in the specification or figures. See also the response to arguments below.

Maintained Claim Rejections - 35 USC § 103

8. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

9. Claims 1, 2 and 5-7 are rejected under 35 U.S.C. 103(a) as being unpatentable over Hauris, U.S. Patent 7,821,183 in view of Blomgren, U.S. Patent 5,781,750

10. As per claim 1, Hauris discloses a microprocessor for processing instructions comprising:

-A plurality of program counters: (Abstract, Column 3, lines 34-35; Column 4, lines 16-41; Figure 2, items 22 and 30)

-A parameter designating a "program counter" (Column 4, lines 16-41, The output from flip flop 46, figure 2 designates a program counter)

-Depending on how the parameter is set, a different relative addressing takes place (Column 4, lines 16-41, The program counters hold different values to access different locations in memory, these values are then incremented when the selected program counter is active. Relative addressing is defined as, "An addressing mode in which the effective address is formed by adding an offset to the program counter (or a portion thereof) during execution." (The Authoritative Dictionary of IEEE Standards Terms) Therefore, incrementing a program counter is relative

addressing, because the program counter has a current value, and a new value is reached by adding an offset to the current PC address.

-Dependent on the parameter, one of said program counters is active in a computation of relative addresses: (Column 4, lines 16-41, The output from flip flop 46, figure 2 designates a program counter)

11. While Hauris does teach a parameter selecting between program counters to access subroutines of instructions, he fails to teach wherein the subroutines are multiple assembler codes.

12. Blomgren teaches:

-Multiple assembler codes being processed: (x86 instruction set and PowerPC instruction set): Blomgren teaches that when a complex x86 instruction is reached, a new instruction pointer is loaded to indicate a software subroutine made up of PowerPC instructions. (Col. 7, lines 35-40)

13. It would have been obvious to one of ordinary skill in the art to combine the invention of Hauris with the multiple assembly codes of Blomgren because as Blomgren states, "Since there is so much installed x86 code, it is greatly desired to run x86 programs on newer RISC CPU's, without the performance degradation of a software emulator," column 3, lines 35-38. Blomgren does this by using a RISC and a CISC decoder, however, for the more complex x86 instructions, he uses a different instruction pointer to access a subroutine of RISC instructions in a ROM. Hauris already teaches accessing a ROM for a subroutine using multiple program counters. It would have been obvious to have the processor decode and execute both RISC and CISC instructions as taught in Blomgren by

having decoders for both instruction sets, and using the ROM subroutine accessing method of Hauris to process the more complex CISC instructions. This would cause said parameter of Hauris to indicate to switch assembly codes because the processor would be in a CISC mode until reaching a complex CISC instruction, at which point a RISC subroutine would be accessed via another program counter.

14. As per claim 2, Hauris, in view of Blomgren, disclose a microprocessor comprising:

-A computation unit: (Hauris, Figure 2 and column 3, lines 20-23, ROM/PLA 16 takes inputs and computes an output for CSDR 24)

-A multiplexer connected to said program counters (Hauris, Figure 2, item 32):

-Said multiplexer receives and is controlled by the parameter: (Hauris, Figure 2 shows the parameter as the input to items 22 and 30 into the CNT input and to the multiplexer input 2, the operation of the parameter is disclosed in column 4, lines 5-41; It is stated that the parameter turns on one program counter and turns off another when it designates a "subroutine call" or "subroutine return" instruction. CNT is part of the decoded parameter.

-Said multiplexer having an output connected to said computation unit for the relative addresses (Hauris, Figure 2, MUX 32 is directly connected to ROM/PLA 16, which is for the relative addresses)

15. As per claim 5, the hardware taught in regards to claim 1 implements the method of claim 5 and therefore, claim 5 is rejected for the same reasons set forth in regards to claim 1 above.

16. As per claim 6, Hauris, in view of Blomgren, teaches claim 5 as shown above, and further discloses:

-Performing one of an addition and a subtraction of an instruction length to/from a program counter reading for a relative address computation in dependence on one of the various operating states and the assembler codes: (Hauris, column 3, lines 31-40 and column 4, lines 16-41 and Abstract)

-Leaving the program counter reading unchanged: (Hauris, column 3, lines 31-40 and column 4, lines 16-41 and Abstract, When a program counter is not selected it remains unchanged. Also, between clock cycles a selected program counter's reading remains unchanged, because the program counter increments periodically, therefore there are periods when the program counter is constant and not incrementing.

17. As per claim 7, Hauris, in view of Blomgren, teaches claim 6 as shown above, and further discloses:

-Performing one of an addition and a subtraction of the instruction length to/from an offset value used for the computation of the relative addresses in dependence on one of the various operating states and the assembler codes: (Hauris, column 3, lines 31-40 and column 4, lines 16-41 and Abstract; The

values stored in the program counters are offsets, and are used for computation of relative addresses.)

-Leaving the offset value unchanged: (Hauris, column 3, lines 31-40 and column 4, lines 16-41 and Abstract; When a program counter is not selected, it's value (the offset) remains unchanged. Also, between clock cycles a selected program counter's reading remains unchanged because the program counter increments periodically, therefore, there are periods when the program counter is constant and not incrementing.)

18. Examiner also notes that as per claims 6 and 7, applicants claim, "performing one of:" and then lists two options multiple times. Therefore, only one of the two options must be met for a reference to teach the claims' limitations.

19. Claims 3 and 4 are rejected under 35 U.S.C. 103(a) as being unpatentable over Goetz, U.S. Patent 5,854,913 in view of Yoshida, U.S. Patent 5,088,030, Baror, U.S. Patent 4,926,323, The PowerPC Architecture, 1994 and K. Short, Embedded Microprocessor Systems Design, 1998.

20. As per claim 3, Goetz discloses:

- A microprocessor for processing various assembler codes: (Abstract, line 1)

- Depending on how the parameter is set, a different relative addressing takes place: (Relative addressing is defined as, "An addressing mode in which the effective address is formed by adding an offset to the program counter (or a portion thereof) during execution." (The Authoritative

Dictionary of IEEE Standards Terms) Therefore, incrementing a program counter is relative addressing, because the program counter has a current value, and a new value is reached by adding an instruction length to the current PC address. Since the Q-bit indicates different instruction lengths to be added to the current PC address, different relative addressing takes place dependent on the Q-bit (Figure 9, Column 16, lines 47-64). Column

- A program counter (NIFA Compute 807, figure 9 and column 16, lines 47-64)

- A computation unit for computing relative addresses: (NIFA Compute 807, figure 9 and column16, lines 47-64)

21. While Goetz does teach multiple instruction sets being implemented and indicated by a parameter, Goetz is silent on how the different offsets for branch instructions are handled. It is well known in the art that PowerPC branch instructions add an offset to the address of the branch instruction (Page 36, numeral 1, The PowerPC Architecture), while x86 branch instructions add an offset to the address of the instruction following the branch instruction (Short, Embedded Microprocessor Systems Design, page 190, 2^{nd} paragraph). However, since Goetz is silent on how the above issue is resolved, Goetz fails to teach:

- A multiplexer having a first input a second input for receiving a zero value, a third input receiving a parameter designating a respective assembler code, a memory for storing an instruction length having an output connected to said first input of said multiplexer

- An addition unit connected between said program counter and said computation unit, said adding unit having a first input connected to said program counter, a second input for an instruction length, and an output connected to said computation unit:

22. Yoshida teaches hardware to implement an x86-like branch instruction, which adds an offset to the address of the instruction following the branch instruction:

- A program counter (register 13, figure 2; Column 4, lines 8-20)

- A computation unit for computing relative addresses: (2nd Adder 17, figure 2 and column 4, lines 1-20)

- An addition unit connected between said program counter and said computation unit: (1st Adder 15)

-Said adding unit having a first input connected to said program counter, a second input for an instruction length, and an output connected to said computation unit: (Figure 2, the 1st adder 15 has the program counter (register 13) as an input, has an input for an instruction length ("Word Length"), and has an output connected to said computation unit (2nd Adder 17).

Yoshida teaches that the branch offset is calculated by adding an offset to the address of the instruction following the branch instruction. This occurs by adding a "Word Length" that is part of the instruction decoded (Figure 2 and column 4, lines 1-20)

23. It would have been obvious to add the hardware of Yoshida to implement the branch target address computation because hardware is inherently necessary in order to carry out the x86 relative branch instructions and because the invention of Yoshida eliminates time required by conventional processors to do branch target calculations (Abstract).

24. While Goetz, in view of Yoshida, teaches a method for handling the x86 branch instruction, it is still necessary to have hardware to implement the PowerPC branch instruction. One of ordinary skill in the art would have recognized that since the purpose of the 1st adder of Yoshida is to add the length of the branch instruction in order to complete the branch instruction that adds an offset to the address of the instruction following the branch instruction, it would have been obvious to simply add a zero as said "Word Length" offset to calculate a PowerPC branch target address. However, Goetz in view of Yoshida fails to teach that the "Word Length" values are stored in a memory and selected via a multiplexer with said parameter as an input.

25. Baror teaches hardwired values, an instruction length and a zero, as an input to an adder to be added to a program counter. Figure 7 shows the multiplexers 712, 716 and 719, which are used to select the offset to be added to the PC value selected from MUX 703. The PC value has one instruction length, a constant value inherently stored in memory, added to it (Column 17, lines 5-15) or can have a zero added to it (Column 17, line 36), see also Column 16, lines 25-28.

26. It would have been obvious to use a multiplexer to select a zero or an instruction length stored in a memory, and to select between the two for adding to the program counter to calculate a target branch address. One of ordinary skill in the art would have recognized that storing the instruction length ("Word Length") in the branch instruction takes up valuable instruction bits and increases the size of instructions and therefore the program size as well. It would have been obvious to use a multiplexer as taught in Baror, to reduce program size or to provide extra bits with which to encode other data or commands.

27. Since Goetz already teaches a parameter indicating which instruction set's offset to add to the program counter for sequential instruction fetching, one of ordinary skill in the art would have recognized to use the same parameter to indicate which offset to add for different non-sequential instruction fetching, i.e., the branch instructions of each instruction set, in order to avoid redundant hardware.

28. As per claim 4, Goetz, in view of Yoshida and Borar fail to teach the only limitation different from claim 3, which is "a subtracting unit" instead of an "adding unit."

29. However, Examiner takes Official Notice that numbers, including offsets, are very frequently represented in two's complement form, which allows simplified binary arithmetic operations.

30. It would have been obvious to one of ordinary skill in the art at the time the invention was made to have the offsets represented in two's complement form

since Examiner takes Official Notice two's complement form allows simplified binary arithmetic operations.

31. It is inherent that a binary adder is also a subtraction unit if the binary inputs are in two's complement form, because there is no difference in hardware between adding two's complement numbers and subtracting two's complement numbers. A two's complement adder is inherently a subtraction unit as well.

Response to Arguments

32. Applicants' arguments filed on 5/13/2005 have been fully considered but they are not persuasive.

33. Applicants argue the 35 U.S.C. § 112 first paragraph rejection of claims 6 and 7.

> "The instant application does describe an embodiment wherein a plurality of program counters is provided and the correct program counter is selected and provided to the proper computation unit"
>
> "Applicants believe that the instant application has precisely disclosed, in a single embodiment, the use of multiple program counters, each having the capability to selectively add or subtract an instruction length or to not add or subtract an instruction length to the program counter"
>
> "The specification of the instant application clearly discloses, that Applicants' disclosed the system of page 4, line 1- page 5, line 5, can, both, add or subtract the instruction length to or from the program counter reading and add or subtract an instruction length to or from the offset value, in dependence on various operating states or assembler codes."

34. These arguments are not found persuasive for the following reasons:

a. To clarify, applicants' attention is directed towards the cited portions of the specification from Applicants' remarks, i.e., page 4, line 1 to page 5, line 5 and also to figures 1-3. Page 4, lines 1-12 describe the embodiment shown in figure 1, including multiple program counters, and selecting the

desired one via a parameter. This embodiment of the invention performs the function set out to be performed by the invention, that is, to perform a different relative addressing dependent on the "assembler code", i.e., instruction set, that is currently being executed. Page 4, lines 14-21 discloses a different embodiment, which uses ***only one*** program counter, not two or more program counters, and also uses an adding unit, in order to perform desired function of the invention. Page 4, line 23 to page 5, line 5 teaches an embodiment, which again, uses ***only one*** program counter, and also a subtracting unit, to perform the desired function of the invention. Page 5, line 18 to page 6, line 4 reiterates the above teachings. There is no single embodiment that teaches a ***plurality of program counters***, each able to ***add or subtract*** in Applicants' cited portions. To clarify again, figure 1 shows two program counters, with no adding or subtracting unit, this is the only figure to show multiple program counters. Figures 2 and 3 each show an adding and subtracting unit respectively, but each only has a single program counter.

35. Applicants argue the novelty/rejection of claims 1 and 5.

"Neither Hauris, nor Blomgren, disclose, among other limitations of Applicants' claims, 'a computation of relative addresses' based on a particular assembler code/operating state of various assembler codes/operating states."

36. These arguments are not found persuasive for the following reasons:

b. To clarify, Applicants' attention is directed towards the 35 U.S.C. 103 rejections to claims 1 and 5 above. Hauris teaches a computation of

relative addresses based on a particular parameter, that is, dependent on the flip flop 46's output (figure 2 and col. 4, lines 16-41), a different program counter is selected. As described above, every instance of incrementing a program counter is a form of relative addressing, as defined by the Authoritative Dictionary of IEEE Standards Terms. Therefore, when a program counter is selected and is incremented, one type of relative addressing occurs, i.e., program counter uPC1, for example, has an offset added to it's current value, when a different program counter is selected and is incremented, a different type of relative addressing occurs, i.e., program counter, uPC2 for example, has an offset added to it. Examiner admitted, as Applicants pointed out, that Hauris does not teach the parameter and different program counters being related to multiple assembler codes (instruction sets), Hauris teaches them being related to different sections of code

c. Blomgren, however, teaches loading an instruction pointer with a value to point to a section or microcode containing a different assembler code than a currently executing instruction's assembler code type. [Col. 7, lines 35-40).] Blomgren teaches that the processor is capable of executing RISC instructions using a RISC decoder, simple CISC instructions using a CISC decoder, and finally, complex CISC instructions via emulation using a microcode sequence of RISC instructions. [Abstract] As stated in the 35 U.S.C. 103 rejection above, it would have been obvious to combine the teachings of Blomgren in the multi-program

counter system of Hauris. Thus, when a microcode section of RISC instructions is accessed in order to implement a complex CISC instruction, the parameter in Hauris would be indicating a different assembler code and also a different relative addressing.

37. Applicants argue the novelty/rejection of claims 1 and 5.

"Applicants' respectfully disagree with the allegations made in the Office Action, and more particularly: a) disagrees that Blomgren shows "a different instruction pointer to access " different assembler codes; and b) further disagrees that combining Blomgren with Hauris, would teach or suggest Applicants' invention of claims 1 and 5."

38. These arguments are not found persuasive for the following reasons:

d. To clarify, Applicants' attention is directed towards Blomgren, col. 7, lines 36-40, "When entry to emulation mode is requested, entry point block 56 generates the proper entry point vector or address in the emulation portion of memory, and loads this address into the instruction pointer 34." This clearly teaches that "when a complex x86 instruction is reached, a new instruction pointer" is "loaded to indicate a software subroutine made up of PowerPC instructions" as Examiner stated in the Office Action of January 13, 2005, and as cited by Applicants on page 15 of the current amendment. Blomgren teaches three modes of operation, a first ***CISC*** mode, a second RISC mode, and a third ***emulation mode*** (which uses RISC instructions). Therefore, different assembler codes are indeed processed (CISC and RISC), and a different instruction pointer (address for microcode subroutine) is loaded to access the different assembler code (RISC).

39. Applicants further argue the novelty/rejection of claims 1 and 5.

"In Blomgren, does not use relative addressing to choose between (i.e., point between) various different assembler codes and/or operating states simultaneously contained in memory, with which to perform an operation, as required by Applicants' claims 1 and 5"

40. These arguments are not found persuasive for the following reasons: In response to Applicants' argument that the references fail to show certain features of Applicants' invention, it is noted that the feature upon which Applicants rely (i.e., the "use of relative addressing to choose between various different assembler codes and/or operating states.") is not recited in the rejected claim(s). The claims require that "a parameter designates" the "different relative addressing" and "respective assembler code", not vice-versa as Applicants are arguing (i.e., that the relative addressing chooses). Examiner also notes that there is no limitation requiring, "various different assembler codes and/or operating states simultaneously contained in memory" in either claim 1 or 5, therefore these arguments are moot. Although the claims are interpreted in light of the specification, limitations from the specification are not read into the claims. See *In re Van Geuns*, 988 F.2d 1181, 26 USPQ2d 1057 (Fed. Cir. 1993). Examiner also notes that Blomgren does teach various assembler codes simultaneously stored in instruction memory. The I Fetch 32 is indexed, via the IPTR 34, to fetch instructions of both the CISC type and RISC type. Also, figure 3 shows the main memory space containing, simultaneously, RISC and CISC instructions.

41. Applicants further argue the novelty/rejection of claims 1 and 5.

"Rather, as taught in col. 5 of Blomgren lines 58-61:

> 'A software routine will be executed that breaks the complex CISC instruction down into several smaller RISC instructions, such as Loads and Stores.'

As such, Blomgren does not use relative addressing to process instructions in one of a plurality of stored various assembler codes."

"Instead, Blomgren breaks down operations from different command sets into a single command set (i.e. RISC), and uses that single command set to process instructions, regardless of the actual command and its particular original command set. This single command set is used to process everything in Blomgren with the disclosed hardware. If there is no command in the single command set (i.e., RISC) of hardware to address a particular command, as disclosed in col. 5 of Blomgren, line 55-58," [emulation mode is entered].

"Thus, Blomgren fails to teach or suggest the computation of relative addressing based on a particular assembler code/operating state of various assembler codes/operating states. Rather, Blomgren only uses the RISC command set to process instructions"

42. These arguments are not found persuasive for the following reasons: To clarify, Applicants' attention is directed towards the abstract of Blomgren, as well as col. 4, lines 34-60. Blomgren teaches three modes of operation, CISC, RISC and complex CISC, which consist of two different assembler codes, RISC and CISC. In the RISC mode and Complex CISC mode, RISC instructions are fetched, decoded and executed. However, in the case of the CISC mode of operation, CISC instructions are fetched, decoded and executed. It is unclear what applicants are attempting to argue. If Applicants are implying that only RISC instructions are present and are fetched, decoded and executed, the teachings cited above refute this argument. If Applicants are implying that Blomgren does not teach the limitations of claims 1 and 5 because there is only one execution unit present to execute both types of assembler codes after they are fetched and decoded separately, then Applicants are arguing a limitation not found in either of the claims 1 or 5.

43. Applicants argue the novelty/rejection of claims 3 and 4.

> "The cited references fail to teach or suggest, among limitations of Applicants' claims 3 and 4, processing various assembler codes, including a multiplexer 'receiving a parameter designating a respective assembler code and, depending on how the parameter is set, a different relative addressing takes place.'"

44. These arguments are not found persuasive for the following reasons: To clarify, Applicants' attention is directed towards the 35 U.S.C. 103 rejections to claims 3 and 4 above.

45. Applicants further argue the novelty/rejection of claims 3 and 4.

> "Applicants respectfully disagree with the allegation that Goetz teaches multiple instruction sets being implemented and indicated by a parameter, in a manner applicable to Applicants' claims 3 and 4. Rather, instead of using Applicants' claimed relative addressing process, including the claimed multiplexer, program counter and computation unit, Goetz offers a much different solution. Contrary to Applicants' invention, Goetz provides two separate command set architectures running in parallel, and uses memory management hardware to switch between them.

46. These arguments are not found persuasive for the following reasons: In response to Applicants' argument that the references fail to show certain features of Applicants' invention, it is noted that the feature upon which Applicants rely (i.e., the limitation requiring that two separate command set architectures are ***not*** run in parallel) is not recited in the rejected claim(s). Although the claims are interpreted in light of the specification, limitations from the specification are not read into the claims. See *In re Van Geuns*, 988 F.2d 1181, 26 USPQ2d 1057 (Fed. Cir. 1993). The nearest limitation in claim 3 states, "a parameter designating a respective assembler code." This is taught by Goetz, see figure 7, the parameter Q designates the respective assembler code (Full x86 and Full PPC).

47. Also, Applicants' attention is directed towards the following teachings of Goetz which show two separate "assembler codes" not running in parallel, col. 6, lines 1-15, col. 8, lines 39-65, col. 9, lines 12-32, and col. 14, lines 1-54. The separate instruction sets are different architectural contexts, which are under control of the mode bit Q. Dependent on the mode, the processor executes instructions in a different manner, i.e., either x86 or PowerPC.

48. Applicants further argue the novelty/rejection of claims 3 and 4.

> "Goetz teaches away from Applicants' claimed invention of claims 3 and 4." [As stated in col. 5 of Goetz, lines 21-43]
>
> "More particularly, using the teachings of Goetz, each of the two architectures of Goetz would contain its own program counter (i.e., two program counters each having a different relative addressing), used to jump to/activate the next command address in the particular architecture's command set, when that particular architecture is enabled (i.e., switched to). Thus, Goetz neither teaches, nor suggests, Applicants' claimed particularly claimed 'program counter' or its particularly claimed use in connection with an adding/subtracting unit for calculating between (i.e., pointing between) the address for the various different assembler codes in a memory."

49. These arguments are not found persuasive for the following reasons: To clarify, Applicants' attention is directed towards figure 9 and the above 35 U.S.C 103 rejection to claims 3 and 4 above. Applicants appear to be speculating as to what Goetz would teach regarding program counters for different instruction sets/relative addressing. However, this aspect of Goetz should not be speculated since Goetz clearly teaches using a single program counter (NIFA Computer 807) for both instructions sets in figure 9 and col. 16, lines 47-64.

50. Examiner also notes that Applicants are only discussing one reference in a 35 U.S.C. 103 rejection containing multiple references. Examiner already

admitted that Goetz does not teach the program counter with its "claimed use in connection with an adding/subtracting unit for calculating between (i.e. pointing between) the address for the various different assembler codes in a memory." However, Examiner stated that Goetz, in view of Baror U.S. Patent 4,926,323, Yoshida, U.S. Patent 5,088,030 The PowerPC Architecture (1994) and K. Short, Embedded Microprocessor Systems Design, 1998, did teach all aspects of claims 3 and 4. In response to Applicants' arguments against the references individually, one cannot show nonobviousness by attacking references individually where the rejections are based on combinations of references. See *In re Keller*, 642 F.2d 413, 208 USPQ 871 (CCPA 1981); *In re Merck & Co.,* 800 F.2d 1091, 231 USPQ 375 (Fed. Cir. 1986).

Conclusion

51. **THIS ACTION IS MADE FINAL.** Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

The following is text cited from 37 CFR 1.111(c): In amending in reply to a rejection of claims in an application or patent under reexamination, the applicant or patent owner must clearly point out the patentable novelty which he or she thinks the claims present in view of the state of the art disclosed by the references cited or the objections made. The applicant or patent owner must also show how the amendments avoid such references or objections.

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is

filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Kevin P Rizzuto whose telephone number is (571) 272-4174. The examiner can normally be reached on M-F, 8-4:30.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Eddie Chan can be reached on (571) 272-4162. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

KPR

Eddie Chan

EDDIE CHAN
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100